Booster Waveform Data

Problem analysis
Tue, Apr 15, 2003

A wrinkle showed up *vis-a-vis* plotting waveforms from Swift digitizers on the Macintosh parameter page. Two signals were being plotted via the Memory Plot option, each from a different node. The data return time was clock event `0x10`-based, so as to capture beam pulse data. The effect was that the waveforms did not appear correlated; they did not often reflect beam cycles. For these BLM front ends, the hardware waveform buffers are digitized at 15 Hz, so that means there is non-beam cycle data that can sometimes be found in the buffers.

Here is the explanation. For these two BLM nodes, the "micro-P start" time is 45 ms after Booster reset event time, so that they begin their cycle activities after Booster extraction, which is about 35 ms after reset event time. The Macintosh client program in this case uses Classic protocol, sending a server-type request to the first signal being plotted. Consider that the server request is sent to `node06C3`, requesting waveform data from both `node06C3` and `node06C6`. Classic protocol support separates out the fulfillment of local data in a request from that of foreign data. So, when the foreign `node06C6` starts its cycle at 45 ms and finishes updating its data pool, it copies out the waveform data and sends it to the server `node06C3`. The server, at the same time when it has updated its own data pool, does nothing about fufilling this request that was sent by itself. Rather, it waits until the server deadline time of 40 ms past the start of its own cycle to fulfill its own part of the request and return both the earlier-received waveform from `node06C6` plus its own to the original client.

Let us examine the timing closely. The server deadline time for any node is 40 ms past the micro-P start time. This means that for the server node, this deadline occurs about 17 ms into the *next* cycle, given that 15 Hz cycles last about 67 ms. At that time, the local waveform data will be copied out of the live buffer, which is by that time being filled with 12.5 KHz digitized data for the cycle following the beam cycle. One could easily see some current non-beam data mixed with some beam data from the previous (beam) cycle.

The effect is not observed in Acnet. The reason is that the support for `RETDAT` or for `FTPMAN` does not try to separate out the local part of the request from the rest. When the server node forwards the entire request, it expects to receive the same forwarded request, but this time it is treated as a nonserver request. (It does not even notice that it was a request sent from itself.) So the timing of when the server node captures the waveform data and when the foreign node captures its waveform data are the same; in this case, they will both capture it soon after their micro-P start time of 45 ms past reset event time, when the waveform buffers will still have the beam data of interest. The foreign node delivers its part at that time, and the server node delivers its part to itself. When the server node receives each contribution, it copies it into the appropriate part of the full reply buffer. At server deadline time, the server node merely ships out the reply buffer that contains both its own as well as the other node's waveform data. In both cases, the live waveform buffer is captured at about the same time.

The effect also would not occur for a case in which the timing of micro-P start was more like that of the Linac, which uses 3 ms for the purpose. In that case, the 40 ms server deadline time occurs well within the same 67 ms cycle.

The effect would not happen for cases in which only beam cycles were digitized. But it would still be wrong, because one might get a mixture of data from two successive beam cycles.